REMARKS

Applicant has considered the Office Action mailed on September 4, 2003, and the references cited therewith. This response cancels claims 6, 10, and 12-30 without prejudice, amends claims 1 and 11, and adds new claims 31-54; as a result, claims 1-5, 7-9, 11, and 31-54 are now pending in this Application.

The title has been changed to reflect the claimed subject matter more clearly, as required in the Office Action.

Affirmation of Election

Applicant affirms the election of claim 1-24 for further prosecution at this time, and has canceled claims 25-30 without prejudice against their subsequent introduction into another application.

Rejections of the Claims under Schroter

Claims 1-24 were rejected under 35 USC § 102(e) as being anticipated by Schroter et al. (U.S. 6,401,192). Claim 22 was rejected under 35 USC § 103(a) as being unpatentable Schroter

Applicant proposes a generalized data processor in which non-essential code that optimizes or aids the execution of essential application-program code is never mixed with the application code, even though the same processor hardware or microarchitecture ultimately processes the instructions of both types of code. The Schroter patent teaches the opposite: a processor that intermixes application code with specialized cache-prefetch optimization instructions, and processes these instructions in a unit (202, Fig. 2) that is entirely separate from those (206-210) that process the code for carrying out the application program.

Claim 1 clearly displays the divergence between these two opposite concepts. "Essential code" and "non-essential code" reside in separate "first" and "second" pipelines, whereas Schroter's essential application code and non-essential prefetch instructions arrive in a single stream to "instruction unit/ branch unit" 204, Fig. 2. Claim 1 further recites a single

microarchitecture structure[1] to process both "code from the first pipeline, and ... code from the second pipeline." In Schroter, "[p]refetch unit 302 executes software 'hint' instructions, thereby performing software initiated prefetches" (col. 3:66-67[2]), but does not process any essential application code, as may be seen from Fig. 2.

Further, Schroter has no element corresponding to a mapping table having multiple "triggers to specify respectively different sequences of the non-essential code."[3] This has no correspondence to the "plurality of prefetch specification data values [that] are loaded into a register," as stated on page 4 of the Office Action. Prefetch values are not code sequences, and they exist for an entirely different purpose. Certainly Schroter does not switch a single microarchitecture from a first pipeline to a second pipeline "in response to the triggers."

Dependent claims 2-5, 7-9, 11, and new claims 31-32 incorporate all the recitations of claim 1. They distinguish the Schroter patent for the same reasons, and for additional reasons as well. For example, in the combination of claims 2 and 3, Applicant is unable to find both a "first instruction cache" and a separate "second instruction cache" particularized for 'providing "hints"in the units named in paragraph 14 of the Office Action; Schroter seems to have but a single cache memory, 214.His multi-level cache L0 (Fig. 3A), 214, 316, 318 (Fig. 3B) all store exactly the same types of instructions. The same is true as to claim 5, discussed in paragraph 16 of the Office Action. As to claim 7, "map[ping] a trigger to a non-essential code sequence" has nothing to do with recording the types of instructions so as to assign them to a correct pipeline, as urged in paragraph 18. As explained above, Applicant's processor need not tell the difference between essential and non-essential instructions because---in contrast to Schroter—they are never intermixed with each other. As to claims 8 and 9, paragraphs 19 and 20 of the Action are

[1] -- Microarchitecture is "the internal design of a microprocessor. Not to be confused with the processor's architecture, which defines its compatibility. Two chips with different microarchitectures but the same architecture can run the same software." (*Tom's Computer Dictionary*, Tom R. Halfhill, 2003, URL http://www.halfhill.com/terms.html, as of December 1, 2003. Practitioners employ this term to refer to the actual low-level hardware for processing instructions, as well as to its design; see, e.g., any of the ACM/IEEE compendia such as the *31st Annual ACM/IEEE International Symposium on Microarchitecture,* IEEE Comput. Soc, Los Alamitos, CA, USA, 1998.

[2] -- The notation " col. nn:nn" indicates column and line numbers in a patent. Thus "col. 2:66-67" refers to col. 2, lines 66-67.

[3] -- The Specification employs the term "sequence" to designate a portion of non-essential code that performs a particular function. See, e.g., page 9:15-22.

silent as to why any entity of the reference might be considered "atomic" (single component) or "vector" (multi-component).

New claims 31 and 32 recite the "dynamic code analyzer" discussed on page 7:20-21 of the Specification, and its creation of "directed acyclic graph trace representations" of the essential code, described on page 8:8-14. Schroter has no analogous unit or corresponding function.

Claims 33-48 present methods according to the present invention. Independent claim 33 loads the essential and non-essential code into different memories,[4] without intermixing them as Schroter does. Claim 33 then stores the mapping table,[5] whose "triggers" and related "code sequences" of non-essential code distinguish Schroter as discussed in connection with claim 1. Claim 33 processes essential code in microarchitecture hardware ("structure"), but, after detecting a trigger, processing non-essential code "in the same" microarchitecture structure as for the essential code. Again, Schroter processes essential application code in one microarchitecture, 206-210, while processing hint code for prefetches in another, control-state machine 203. Further, Schroter's different prefetch data values do not correspond to "non-essential code sequences."[6]

Dependent claims 34-48 incorporate all the features of parent claim 33. Claims 34-37 detail the memories holding the two separate code streams as pipelines,[7] or caches which can be only logically separated[8] or physically separated as well.[9]

Schroter's non-essential or hint code performs only a single function: prefetching instruction code. As recited in claim 39, Applicant's non-essential code performs this function,[10] but can also test the microarchitecture,[11] perform sandbox or other security checks,[12] perform

---

[4] -- Specification page 5:9-11.
[5] -- Specification, page 9:23-27.
[6] -- Specification, page 9:27-29.
[7] -- Pipelines are inherently staged buffer memories. See, e.g., *Computer Dictionary Online*, at URL http://www.computer-dictionary-online.org/?q=pipeline, "pipeline <architecture> A sequence of functional units ('stages') which performs a task in several steps, like an assembly line in a factory. Each functional unit takes inputs and produces outputs which are stored in its output buffer. One stage's output buffer is the next stage's input buffer...." (accessed December 1, 2003)
[8] -- Specification, page 5:9.
[9] -- Specification, page 5:11-13.
[10] -- Specification, page 5:14-17.
[11] -- Specification, page 14:10-14.
[12] -- Specification, page 9:12-13, page 15:13-14.

speculative execution,[13] process interrupts or exceptions,[14] or virtualize instructions or instruction sets.[15] Schroter does not suggest anything except prefetching, and shows no mechanism for accomplishing any of these tasks, much less multiple tasks, as recited in claim 40.

Dependent claim 41 sores certain of the code in the second memory along with non-essential code, as described on page 4 lines 9-12 of the Specification. Claim 42 may employ such code for "virtualization." Schroter, of course, has no suggestion of virtualizing anything. Claim 43 names some of the many entities that can be virtualized: "individual instructions,"[16] "blocks of instructions,"[17] "sets of registers,"[18] and "processor hardware resources"[19] (some f these may overlap).

Dependent claims 44-48 delineate some classes of triggers envisioned in the Application. Again, Schroter has only one restricted type of entity that can trigger his prefetch unit---an instruction in the stream of essential application code. Applicant' stable of triggers is much more general. "Instruction attributes"[20] may serve as triggers (claim 44), and those attributes may include (Claim 45) "opcodes, locations, and operands"[21] in any combination. Claim 44 includes "data attributes,"[22] such as data "values and/or locations"[23] (claim 46). "State attributes,"[24] in claim 44 may include "architectural and/or microarchitectural states"[25] (claim 47). "Event attributes"[26] may include "interrupts, exceptions, and/or processor state register values"[27] (claim 48). The different attributes and classes may be combined with each other, as noted in claim 44.

Claims 49-55 delineate systems according to the invention. Schroter has no separate "first and second pipelines" for preventing the intermixing of essential and non-essential code, as in independent claim 49. Nor does he teach a single microarchitecture structure coupled to both

---

[13] -- Specification, page 9:21.
[14] -- Specification, page 16:8-10.
[15] -- Specification, page 16:23-30.
[16] -- Specification, page 10:12-18, 24-30; page 17:24-26.
[17] -- Specification, page 18:13-20.
[18] -- Specification, page 12:13-24.
[19] -- Specification, page 13:1-2.
[20] -- Specification, page 11:8-13:16.
[21] -- Specification, page 11:10-11.
[22] -- Specification, page 13:18-15:23.
[23] -- Specification, page 13:20-21.
[24] -- Specification, page 15:25-16:2.
[25] -- Specification, page 15:27.
[26] -- Specification, page 16:4-21.
[27] -- Specification, page 16:6, 11-12.

pipelines for processing both essential code and sequences of non-essential code. Finally, Schroter has no memory coupled to both the pipelines to store the essential and the nonessential code "as separate parts of the same library."[28]

Dependent claims 50-54 elaborate on claim 49. Claim 51 specifies the memory type as "hard disk, a floppy disk, RAM, ROM, a flash memory, and/or a medium readable by a machine."[29] or as " Claim 52 recites "separate sections of the same file.[30] Schroter has no such file to store the essential and non-essential code together, yet unmixed. Claims 53-53 place the code in a "static file" or in a "run-time library."[31]

Rejections of the Claims under Watts

Claims 1 and 2 were rejected under 35 USC § 102(b) as being anticipated by Watts et al. (U.S. 5,881,135). A broad purpose of the present invention is to promote the execution of an application program by executing along with it certain performance-enhancement ("non-essential") code that alters the operation of a processor while it is executing the ("essential") code of the application. Watts teaches a speech-processing system having a number of different channels for serving simultaneous calls. Each channel has its own copy of certain application programs. A few channels may occasionally need an enhanced version of one of these programs. The invention calls this enhanced version and substitutes it for the standard version when required.

All of Watts' code is therefore "essential" application code that performs a user function, in the context of claim 1. Watts has none of claim 1's "non-essential" code that runs along with the essential code to aid its execution; see the description of non-essential code at, e.g., page 4:4-6 of the Specification. Further, Watts has nothing similar to the mapping table of claim 1, which contains "triggers to specify different sequences" of the non-essential code. The statement in paragraph 6 of the Office Action incorrectly states that

> "it is inherent that some table or list must exist so that the main processor or some other logic device knows to send the floating point operations to the co-processor, and that it

[28] -- Specification, page 20:12-13.
[29] -- Specification, page 22:9-12.
[30] -- Specification, page 19:17-24.
[31] -- Specification, page 19:17-18 et seq.

would keep all other types of operations for execution in the host, or primary, processor...."

In fact, the main processor detects the presence of a certain voltage on one of the chip's pins; see col. 33:9-12, cited in the Office Action.[32] In addition, the purpose of a hypothesized table in Watts is to separate out the instructions to be executed in one processor from those f rom those to be executed in the other. Applicants store essential and non-essential instructions in different pipelines, and therefore do not require a table for this purpose. In any case, however, conventional processors such as Watts' can allocate instructions to different execution units in a number of ways, including logic decoders to detect distinctive bit patterns in the instruction, prefix instructions attached to coprocessor instructions, and service instructions to switch between a hardware coprocessor and a main-processor emulation of a coprocessor function. Thus, no 'inherent' table can be imputed to Watts.

Claim 2 depends from claim 1, and distinguishes over Watts for the same reasons. In addition, the statement in paragraph 10 that Watts' instruction cache holds "instructions that determine the logical correctness of a program" appears to be circular, because the remainder of the same sentence implies that this attribute applies to *all* instructions in Watts' system.

[32] -- This is in fact the standard procedure for this type of chip, not only for the Motorola 68681.

Conclusion

For the above reasons, Applicant urges that the claims are in condition for allowance, and respectfully requests reexamination and allowance. The Examiner is invited to telephone Applicant's attorney at (612) 373-6971 to facilitate prosecution of this Application.

If necessary, please charge any additional fees or credit overpayment to Deposit Account No. 19-0743.

Respectfully submitted,

HONG WANG ET AL.

By their Representatives,

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
Attorneys for Intel Corporation
P.O. Box 2938
Minneapolis, Minnesota 55402
(612) 373-6971

Date 2 Dec 2003 By _____
J. Michael Anglin
Reg. No. 24,916

CERTIFICATE UNDER 37 CFR 1.8: The undersigned hereby certifies that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to: Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on this 2 day of December, 2003.

KACIA LEE _____

Name

Signature